SOFTWARE ARTS TECHNICAL NOTES, BY THE CREATORS OF VISICALC

A Publication of
Software Arts, Inc.

SATN

THE JOURNAL FOR VISICALC USERS

Vol. I, No. 3
January/February 1982

IN THIS ISSUE:

Date Arithmetic and Calendars 1
The VisiCalc Calendar 1
A General Formula for Date Arithmetic 4
Constructing a Receivables Form 7
Constructing a Scheduling Form 8
Double Your Number 12
Solution to Send More Money 12
. . . Plus VisiCalc Pointers

DATE ARITHMETIC AND CALENDARS

Although often considered strictly a financial tool, the VisiCalc® program can also be used as another common business aid, the calendar. Unlike a printed calendar, though, a VisiCalc calendar is a dynamic planning device.

This issue of SATN will show you how to construct a calendar with the VisiCalc program, and how to use that calendar to manipulate dates.

To do that, we have developed a general approach, or algorithm, that is described in the article, *A General Formula for Date Arithmetic.* A careful reading (and entry on a VisiCalc sheet) of that article is basic to an understanding of the two articles that follow.

After you've entered the algorithm on a VisiCalc sheet you may apply it, as your interests and needs require, to figure out receivable due dates, or develop a schedule with the VisiCalc program.

Among the things that we'll show you are how to create a DIF file to preserve sheet memory and reduce processing time, and how to construct a VisiCalc model that can serve as a template for repeated use. The models we have developed make extensive use of the @IF function, but if your version of the VisiCalc program does not have that feature, we'll show how to simulate it with @MIN and @MAX.

The techniques outlined in the various articles, while made specific to the applications we used, can be applied in other ways. For example, in the scheduling article we show how to calculate the exact number of days between two dates. If you are a financial manager, that can be an important factor in determining the costs of money or rates of return. And although the article on scheduling uses a manufacturing example, it can easily be adapted, in whole or in part, to such things as scheduling sales in retail stores, or even scheduling employee vacations.

THE VISICALC CALENDAR

Rather than giving you a large calendar that will work in the applications to follow, we will look at how to go about actually constructing a VisiCalc® calendar. That way you will be able to save it as a VisiCalc file and later update it.

Constructing A Yearly Calendar

Date arithmetic assumes that time begins at some point equal to zero. For our purposes we have taken that to be January 1, 1981. You can begin with any year you choose, but don't try to go back too far, since until the middle of the 17th century the first day of a new year occurred sometime in March, not at the beginning of January!

Begin by setting the sheet up with the command /GC7 and with /GOR. Next, enter the labels shown in Figure 1. Then enter:

```
>B1:1981
```

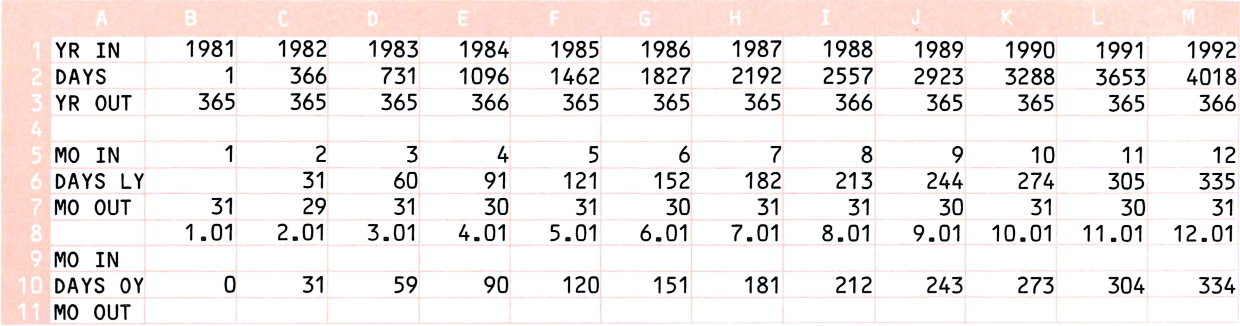
To extend the table to any number of years you wish, go to C1, add 1 to the entry in B1, and replicate that entry relatively across the sheet. Enter:

```
>C1:+B1+1
>C1:/R:D1 . . . M1:R
```

We ended our table at 1992, because in the scheduling article we will use the space to the right of column M. If you choose to extend the calendar further, you may, but remember that the year 2000 is not a leap year because it is evenly divisible by 400.

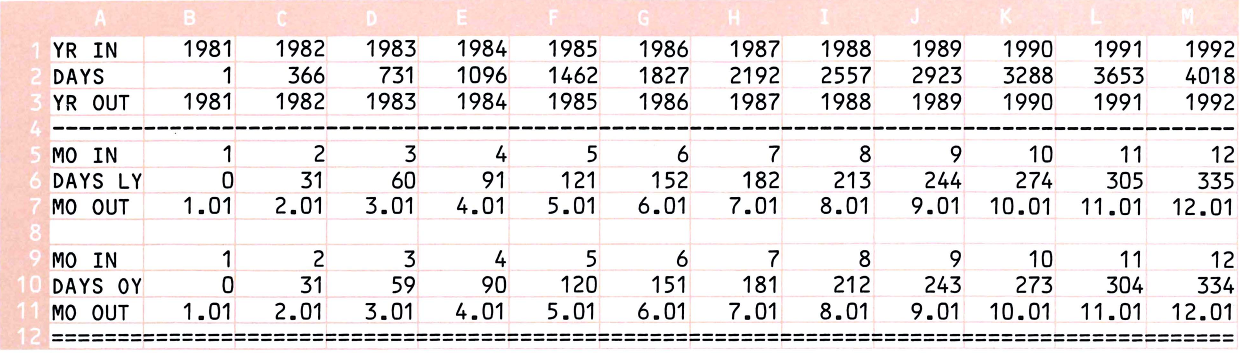
Date arithmetic is a matter of counting days, so we must enter the number of days in each ordinary year, and then account for leap years. Enter:

```
>B3:365®/R:C3 . . . D3:
>E3:366
```



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	YR IN	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	1991	1992
2	DAYS	1	366	731	1096	1462	1827	2192	2557	2923	3288	3653	4018
3	YR OUT	365	365	365	366	365	365	365	366	365	365	365	366
4													
5	MO IN	1	2	3	4	5	6	7	8	9	10	11	12
6	DAYS LY		31	60	91	121	152	182	213	244	274	305	335
7	MO OUT	31	29	31	30	31	30	31	31	30	31	30	31
8		1.01	2.01	3.01	4.01	5.01	6.01	7.01	8.01	9.01	10.01	11.01	12.01
9	MO IN												
10	DAYS OY	0	31	59	90	120	151	181	212	243	273	304	334
11	MO OUT												

FIGURE 1. *The file CALWORK, before it is saved as a DIF file.*



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	YR IN	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	1991	1992
2	DAYS	1	366	731	1096	1462	1827	2192	2557	2923	3288	3653	4018
3	YR OUT	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	1991	1992
4	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------
5	MO IN	1	2	3	4	5	6	7	8	9	10	11	12
6	DAYS LY	0	31	60	91	121	152	182	213	244	274	305	335
7	MO OUT	1.01	2.01	3.01	4.01	5.01	6.01	7.01	8.01	9.01	10.01	11.01	12.01
8													
9	MO IN	1	2	3	4	5	6	7	8	9	10	11	12
10	DAYS OY	0	31	59	90	120	151	181	212	243	273	304	334
11	MO OUT	1.01	2.01	3.01	4.01	5.01	6.01	7.01	8.01	9.01	10.01	11.01	12.01
12	=========	=========	=========	=========	=========	=========	=========	=========	=========	=========	=========	=========	=========

FIGURE 2. *This figure shows what the final version of the VisiCalc calendar should look like after it has been edited and saved as the file CALVALU.*

Now replicate the entries in B3 . . . E3 into the cells for the remaining years. Since the entries we are replicating follow a regular pattern, we can do so with a few keystrokes. Enter:

```
>F3:+B3
>F3:/R:G3...M3:R
```

The next step is to count the number of days from the beginning of the calendar to its end, a matter of accumulating the entries in row 3. In row 2 enter:

```
>B2:1
```

In effect we are saying that we will only count 1 day in 1981. The rest of the days in the year are counted on a month by month calendar.

Now, at C2 we can find out how many days will elapse from the beginning of 1981 until the beginning of 1982. Enter:

```
>C2:+B3+B2
```

That adds the one day we've counted for 1981 to the full number of days in the year, and says that 1982 begins on the 366th day after January 1, 1981. To finish calculating the number of days that will elapse from the beginning to the end of the calendar, replicate the entry in C2 across the sheet:

```
>C2:/R:D2...M2:RR
```

Now, if you scroll over to M2, you'll see that the year 1992 begins 4018 days after January 1, 1981.

Notice that the table on your sheet differs from the one in Figure 2, though it corresponds to the sheet shown in Figure 1. Figure 2 is an illustration of how the calendar will look when you have finished this article, and Figure 1 shows how it will look as you develop the calendar. Rows 1 and 2 are identical, but row 3 on your sheet contains the series 365, 365, 365, 366 . . . 366 as in Figure 1. You cannot erase or delete row 3 because that would destroy the calculations in row 2.

The solution is to eliminate the expressions, and turn the entries into data. Since that is what occurs when a DIF file is saved, we can create and use a DIF file to turn rows 1 and 2 into data. However, we will be performing the same task again soon, so for efficiency's sake we should complete as much of the table as necessary and then perform a single save.

SATN is published bi-monthly by Software Arts, Inc., P.O. Box 494, Cambridge, Massachusetts 02139. Address all editorial correspondence to the Editorial Department, SATN, P.O. Box 494, Cambridge, MA 02139. Requests for special permission or bulk orders should be addressed to the publisher.

Domestic subscriptions: 6 issues, $30.00; outside U.S.A., $50.00. Single copy price, $6.00. Send subscriptions, change of address, and fulfillment questions to SATN Subscriptions, P.O. Box 815, Quincy, MA 02169.

Copyright © 1982, Software Arts, Inc. All rights reserved. This periodical may not be copied or reproduced in any fashion either in whole or in part without the prior written consent of Software Arts, Inc.

Software Arts, Inc. makes no warranty, express or implied, with respect to the quality, accuracy or freedom from error of the information contained in SATN, the journal for VisiCalc® users, and specifically disclaims all liability for any damages resulting from the use of such information.

Apple is a registered trademark of Apple Computer, Inc. Atari is a trademark of Atari Inc. Commodore is a trademark of Commodore International. DIF is a trademark of Software Arts, Inc. HP-83/85 and HP-125 are trademarks of Hewlett-Packard Company. IBM is a registered trademark of International Business Machines Corporation. TRS-80 is a trademark of Radio Shack Division of Tandy Corp.

SATN is a trademark of Software Arts, Inc.
VisiCalc is a registered trademark of Personal Software Inc.

Constructing the Monthly Calendar

To do the arithmetic for months and days requires a table that in most respects is similar to the year table just developed. However, because leap years contain an "extra" day, we must have two tables.

There is no choice of when to start a monthly calendar; it's January. Beginning at B5 enter the numerical equivalents of the months:

```
>B5:1
>C5:+B5+1
>C5:/R:D5 . . . M5:R
```

That creates the series 1 . . . 12 in row 5.

In row 7, enter the number of days in each month as shown in Figure 1. Notice that February shows 29 days, not 28. When we construct the table for ordinary years we will have to subtract 1 from all months after February to adjust for leap year.

When we created the year table we included one day in 1981, and by doing so included January 1 in every year of the calendar. Including January 1 as a "countable" day in the table of months would be to count it twice, making our calculations off by 1 day. Therefore, begin counting days in the months with the entry:

```
>B6:0
```

Now add the number of days in the months, and then replicate it relatively to December by entering:

```
>C6:+B6+B7
>C6:/R:D6 . . . M6:RR
```

The table produced by those entries is for a leap year, and shows the number of the day on which the preceding month ends. To complete the table, we have placed below those days a compressed version of the date on which that day falls. January 1st is compressed to 1.01 and each of the following entries is 1 greater than 1.01. To add that line to the table, enter:

```
>B8:+B5+.01
>B8:/R:C8 . . . M8:R
```

To add the table for ordinary years requires only a few replications of the entries in the leap year table and an adjustment for the shorter month of February. First, enter:

```
>B10:+B6
>C10:+C6
```

At C8 we make the adjustment for leap years by subtracting 1 from the entry at C7, and then replicate that entry through the year. Enter:

```
>D10:+D6-1
>D10:/R:E10 . . . M10:R
```

Your sheet should now look exactly like the one in Figure 1. Though it contains all the essential information for finding any date falling between January 1, 1981 and December 31, 1992, it has to be formatted in a more useful way.

The next step is to change the tables we have created from a series of calculations into tables that contain only data. That allows us to conserve memory space on the sheet and, since no calculations would have to be done on the table, to reduce processing time when we actually enter dates.

Turning Expressions into Data

DIF files contain only data, no expressions or formatting commands. The tables that we shall use need no further calculation or alterations, but data only. Therefore we'll turn the tables we have just created into DIF files, so we can edit them without affecting the calculations.

Before converting the tables, it is a good idea to save the expressions so you can return to the file and update it by starting with a new year at B1. So, save the sheet with /SS, giving it the filename CALWORK.

Having saved the file with the expressions, return the cursor to B1 and enter:

```
>B1:/S#S
```

Give this file the name CALVALU. The prompt line will read DATA SAVE: LOWER RIGHT. Enter:

```
M10
```

KEY TO CONVENTIONS USED IN SATN

The VisiCalc® program is available on personal computers manufactured by Apple, Atari, Commodore, Hewlett-Packard, IBM, and Radio Shack. To reduce any confusion arising from the differences among the keyboards of those computer systems, SATN uses a standard symbol and name to represent certain commonly used keys.

The symbol ® and the word RETURN are used as synonyms for the keys known as ENTER, ENDLINE, and RETURN. The symbol ^ is used to represent an exponentiation operator.

To make VisiCalc documentation more readable, the program was designed so that the colon (:) has the same effect as RETURN in some commands, as when ending the > command or separating various elements of the replicate command.

In examples of what to type, type exactly what you see and press RETURN at the end of every line. Users of the HP-83/85 may find it necessary to add parentheses to some of the expressions included in the text. For example, 3+4*5 should be (3+4)*5 on an HP-83/85.

Throughout SATN, one typeface represents both VisiCalc input and output (what you should type and what you will see on the screen). The typeface looks like this:

```
This line is typeset in OCR-B.
So is this: +A1+B1*3.25
```

To allow you to type the formulas directly on a VisiCalc sheet, examples in SATN specify coordinates, like A1, rather than the names of the values they represent. When actually typing a formula, you can use the arrows to point rather than using the > command.

The status line will now read DATA SAVE: R, C OR RETURN. Save the data in rows. Enter:

```
R
```

After the file CALVALU has been saved, enter:

```
>B1:/S#LCALVALU
```

The prompt line will now read DATA LOAD: R, C, OR RETURN. Once again, enter R. The VisiCalc program will write over all the cells containing the expressions used to count days, and though their appearance will not change at all, the cells now contain only the values that were calculated, not the calculations themselves.

Building the Working Calendar

To make the sheet identical to Figure 2, you have to replicate row 1 into row 3. That is done with the entry:

```
>B1:/R...M1:B3:
```

That entry completes the year calendar.

Since there were no calculated values in row 1, you did not have to specify whether the entry was to be copied relatively or with no change. (And since you replicated a range of entries into another range in the same direction, you only had to specify the beginning of the target range.)

Editing the month table is essentially the same as editing the year table. Row 5 is replicated into row 9 and row 8 into rows 7 and 11; then row 8 is deleted and a blank row inserted. Enter:

```
>B5:/R...M5:B9:
>B8:/R...M8:B7:/R...M8:B11:
>A8:/DR/IR
```

Now save the sheet as a DIF file once more, using the same commands and filename as above. This time, however, position the cursor at A1 before you enter the /S#S command, so the new file will include the labels. When you enter the filename CALVALU you may see the prompt DATA SAVE: FILE EXISTS, Y TO REPLACE. Enter: Y

You will then be prompted for the LOWER RIGHT, which is now at M11, and whether you wish to save by row or column, so enter:

```
M11®R
```

And the final version of the VisiCalc calendar you have created will be saved.

The calendar is now ready for use in any application that requires one. Since it has been saved as a VisiCalc file, CALWORK, containing the original calculations, it can be recalled to the screen and revised (if, for example, you need a calendar that begins with a different year). It has also been saved as a DIF file, CALVALU, that you can load in any sheets that might require calendars.

Keep a record of the coordinates of the lower right of the calendar. Then, if you create a model that requires a calendar you'll know how much space it requires and can provide for it on the sheet you're developing.

A GENERAL FORMULA FOR DATE ARITHMETIC

A VisiCalc calendar can be a practical tool once we develop a general approach, or algorithm, that allows us to put the calendar to work. In this article, we'll describe the VisiCalc expressions that are used to convert such dates as 11.23 1981 into a number of days, add and subtract days from that, and convert the result back into a date. The two articles that follow will be based directly on the sheet that is developed in this article.

Determining when a receivable is due or a project must be started are the same problem: given one date and some number of weeks or days, when will another date fall?

Finding the due date of a receivable means counting all days to the present and adding the number of days you will wait for payment; when finding the start date of a project, you count all days to a future date and subtract as many days as it takes to complete the work. Since adding a negative number is the same as subtracting a positive one, both receivable due dates and scheduling start dates can be calculated with the same general formula.

The first step in each case is the same: you count the number of days from the beginning of time (defined as January 1, 1981) until the given date. The formula, in general terms, is:

Days To Given Year + Days To Given Month +
Days to Date in Given Month = Days To Given Date

Where the Given Date is either the present or some future date.

To Days to Given Date we would add, say, 30 or 60 days, if we were calculating receivables, or add –220 days if scheduling a project. Thus:

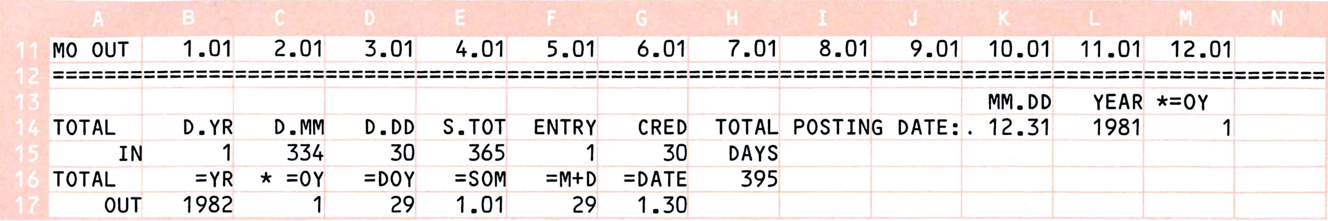
Days To Given Date + Credit (Or Completion Time)
= Date In Days

That formula yields a total number of days that is in fact the date we are seeking. But if you sent someone a bill marked as payable on the 658th day after January 1, 1981 (rather than October 20, 1982), you could reasonably expect to confuse the person. Thus the date in days must be broken down into the month, day, and year. We do that with a series of subtractions, which are described by two general formulas:

Date In Days – Days In Full Years =
Days Till Year Due + X days

X days – Days In Full Months =
Days Till Month Due + Days In Month Due

The real work that the VisiCalc® program can do is to take specific statements of those general formulas and use them to calculate the actual days and dates. We have set up the sheet shown in Figure 3 to serve as the basis for both a billing and a scheduling application. There are a few minor differences between the two applications, but in all important respects the same approach works for both.



```
        A        B        C        D        E        F        G        H        I        J        K        L        M        N
11 MO OUT      1.01     2.01     3.01     4.01     5.01     6.01     7.01     8.01     9.01    10.01    11.01    12.01
12 ========================================================================================================================
13                                                                                            MM.DD     YEAR *=OY
14 TOTAL       D.YR     D.MM     D.DD    S.TOT    ENTRY     CRED    TOTAL POSTING DATE:.    12.31     1981        1
15       IN       1      334       30      365        1       30     DAYS
16 TOTAL        =YR  * =OY     =DOY     =SOM     =M+D    =DATE      395
17      OUT    1982        1       29     1.01       29     1.30
```

FIGURE 3. *The first line of this figure shows the calendar, loaded on the sheet with /S#LCALVALU. Note that the labels in this figure have been formatted with /FR, and that the cells K14 and G17, in which dates appear, have been formatted with /F$, to show the dates to two decimal places.*

Adding Up the Days

To begin, if CALVALU is not on the screen, load it with /S#L. Adjust the column width to 7 with /GC7, and the order of recalculation to rows with /GOR. Next, enter the labels shown in Figure 3 at I14 and J14. Those labels identify the field POSTING DATE:. Then enter December 31, 1981, as follows:

```
>K14:/F$12.31
>L14:1981
```

The first task that the VisiCalc program performs is deciding whether the year is a leap year. Without that information, your calculations could be off by one day. To get the program to pick the right calendar, we've given leap years a value of 0, ordinary years a value of 1, and written an expression to derive those values.

Since a leap year is evenly divisible by 4, dividing an ordinary year by 4 produces a remainder. For example, 1981/4 = 495.25. We are only interested in whether there is a remainder, so we must break off the integer portion (495) of the division. That is done, in general terms, with the expression:

Year/4 – @INT(Year/4) = Remainder

If the year is evenly divisible by 4, the answer is 0. If not, it is .25, .50, or .75. That still does not yield the 1 that identifies ordinary years, but we can generate a 1 or a 0 with the @IF or @MIN functions. First, the @IF function. In general terms @IF says: IF this is true, THEN use that value; if not, use the value of something ELSE.

We specify the truth or falsity of the @IF statement by using one or more of several symbols; greater than (>), less than (<), equal to (=), greater than or equal to (> =), or less than or equal to (< =).

The expression to give the values assigned to the years is:

@IF(Year/4 – @INT(Year/4)>0,1,0)

In plain language that says: If the division yields a remainder (>0), return a 1, and if not, return a 0. At M14, enter:

```
>M14:/F*@IF(L14/4-@INT(L14/4)>0,1,0)
```

We've formatted the command with /F* so that in leap years the cell is blank and in ordinary years it contains a single asterisk, therefore M13 contains the label * = 0Y.

If your version of the VisiCalc program does not have the @IF function, you can use @MIN. The general formula to set the value of cell M14 to 1 or 0 is:

@MIN(Year/4 – @INT(Year/4)*4,1)

Thus if the remainder is .75, it is multiplied by 4, yielding 3, and 1 (the minimum) is returned at M14. If there is no remainder, 0 is returned. The actual entry that simulates the @IF function is:

```
>M14:/F*@MIN(L14/4-@INT(L14/4)*4,1)
```

Note that these expressions will not work for the year 2000, which, though not a leap year, is evenly divisible by 4. If you are planning that far ahead, keep that in mind.

Having established whether the year in question is a leap year, we can begin counting days. To avoid forward references, that is done to the left of the location where we need the final results, and at least one row below the one containing the first variable, the date. Then the date can be changed without having to recalculate with the ! command.

To keep track of each step, enter the labels shown in Figure 3 in row 14. Except for the labels in columns F, G, and H, they are identical for both applications. D.YR means "days to beginning of year"; D.MM, "days to beginning of month"; and D.DD, "days from beginning of month." S.TOT is a subtotal of the days, and in fact, is the number of days from the beginning of time until the first date given.

The first step is to find how many days have elapsed until the current year began. That's done by entering the expression:

```
>B15:@LOOKUP(L14,B1...M1)
```

Next we calculate the days to the beginning of the month. Because we have to select the right calendar, we'll use @IF once again. Enter:

```
>C15:@IF(M14<1,@LOOKUP(K14,B5...M5),
  @LOOKUP(K14,B9...M9))
```

That selects the correct calendar and returns the number of days from the beginning of the year to the beginning of the month in the given date. Note that we could also have written the conditional part of the statement as M14 = 0, since the only values that appear there are 1 or 0.

To simulate the @IF function, use the technique described in SATN, Vol I, No. 1. That is, at C15 you would make an entry that represents the general formula:

(1 – Code) * (@LOOKUP(Days,Leap Yr Range) +
(Code * @LOOKUP(Days,Ord Yr Range))

If the code is 1, an ordinary year, subtracting the code from 1 yields zero. Zero times the leap year table equals zero, to which the ordinary year table is added. Conversely, if the code is 0, then the ordinary year table is made equal to zero and the leap year table is multiplied by 1 – 0, which is 1. On the sheet the entry is:

```
>C15:1-M14*@LOOKUP(K14,B5 . . . M5) +
  (M14*@LOOKUP(K14,B9 . . . M9))
```

Now, to count the days from the beginning of the month we have to break off the month portion of the compressed statement we're using and multiply the remainder by 100 to make it a whole number. If we are finding the due date of a bill, it's customary to begin counting on the day after the posting date. So that we do not include the posting date in the total we must subtract 1 day.

Break the date apart with the @INT function, and then add the days with @SUM. Enter:

```
>D15:+K14-@INT(K14)*100-1
>E15:@SUM(B15 . . . D15)
```

That sum is the total number of days from the beginning of the calendar to the given date.

Beginning in column F, the sheet for billing differs slightly from one for scheduling. We'll proceed as though setting up a sheet for billing, and discuss the differences in the article on scheduling. They do not alter the way new dates are calculated.

Note that the entry at F15 is not a factor in the calculations. Its purpose is to identify the line in the part of the sheet where bills are posted.

We now have the number of days to the posting date and must add the number of days we will wait for payment. Assume that 30 days is a reasonable wait. Beneath the label CRED at G14, enter:

```
>G15:30
```

At H16, CRED is added to the total days with the entry:

```
>H16:+E15+G15
```

That yields the key piece of information, the date, but stated in days, not as we need it.

Converting Days Into Dates

What must now be done is to convert the number of days into a day, month, and year, the reverse of the process used to find the number of days. That is, we begin subtracting.

In row 16, as in row 14, is a series of labels to identify each step. The first, =YR, identifies the location below, where the year is returned. The label * =OY identifies ordinary years; =DOY means "days of year"; =SOM, "start of month"; =+D, "plus days" (from start of month); and =DATE, the date in compressed form. Enter these labels.

Then, to find the year, enter:

```
>B17:@LOOKUP(H16,B2 . . . M2)
```

The range is different, so that a year rather than a number of days is returned, but the entry is essentially the same as the one used to start the count.

Next, as above, we must code leap years. The expression required differs only in the cell to which it refers:

```
>C17:@IF(B17/4-@INT(B17/4)>0,1,0)
```

Or, if you do not have the @IF function:

```
>C17:@MIN(B17/4-@INT(B17/4)*4,1)
```

Now subtract the number of days in the year from the total days with:

```
>D17:+H16-@LOOKUP(B17,B1 . . . M1)
```

That yields the date, stated as days after the first of the year. It is translated into the correct month with the entry:

```
>E17:@IF(C17<1,
  @LOOKUP(D17,B6 . . . M6),
  @LOOKUP(D17,B10 . . . M10))
```

Or, if you do not use a logical function:

```
>E17:1-C17*@LOOKUP(D17,B6 . . . M6)+
  (C17*@LOOKUP(D17,B10 . . . M10))
```

The days from the beginning of the month until the due date are obtained by subtracting the days until the month begins from the balance of days remaining. Enter:

```
>F17:@IF(C17<1,
  D17-@LOOKUP(E17,B5 . . . M5),
  D17-@LOOKUP(E17,B9 . . .M9))
```

Or, if you are not using a logical function, simulate one with the entry:

```
>F17:+D17-
  ((1-C17*@LOOKUP(E17,B6 . . . M6))+
  (C17*@LOOKUP(E17,B10 . . . M10)))
```

At E17 you should show 1.01, and 29 at F17. Those values are compressed into a date at G17 with the entry:

```
>G17:/F$+E17+(F17/100)
```

By formatting the cell with /F$, dates like January 10th display as 1.10, not 1.1.

Now, at B17 and G17 are the year and date we have been looking for. If you have not changed the date entered at K14 and L14, or the 30 at G15, those cells should read 1982 and 1.30 respectively.

Save the sheet you have created, giving it the filename DATEWORK.

The sheet you have just finished constitutes the basis for both the billing and scheduling articles that follow. In both of those articles you will frequently find expressions that differ only slightly from the ones you have just entered. Most of the differences are in the cells referenced, and in many cases the expressions are identical to the ones above. In short, those articles will require very little more in terms of new entries. In fact, in the billing article, you will find that most of what remains to be done is to replicate the expressions that are already on your sheet.

VISICALC POINTER . . .

Sometimes you may not be certain how much space will be required for a series of entries that will become a range. How can you plan for that?

The figures that appear in SATN often show lines of dashes or asterisks that not only add to the appearance of the sheet, but also serve another purpose as the sheet is developed. By including the cells containing the dashes or asterisks in a range, we can provide room to expand the range.

For example, if you have entered a column of figures that will be added with @SUM(A1 . . . A10), and A10 is not an entered value, but a line of dashes, then you can insert (/I) a row at A10 and it will be included in the range automatically. The dashes would be moved to row 11, and the range would read A1 . . . A11. If the range had not included the dashes, but read A1 . . . A9, you could not have inserted at A10 a row that would be included in the range. You can insert a row or a column in a range at other locations, and expand the range that way, but at times the only logical place to add to a range is at its end.

You could leave a blank row or column at the end of the range, of course, but the dashes or asterisks serve as a reminder that the space has been reserved.

This technique works with @MAX, @CHOOSE, @LOOKUP, @SUM, @AVERAGE, and @NPV.

CONSTRUCTING A RECEIVABLES FORM

Most of the work required to set up a receivables form with the VisiCalc® program should now be completed. If you have cleared your sheet, load the file DATEWORK.

Using the repeating label command, /–, enter and replicate a series of lines across the sheet from columns I through N and in rows 18, 21, 24, 27, and 30. Enter:

```
>I18:/--
>I18:/R:I21:/R...I21:I24
>I18:/R...I27:J18...N18
```

As shown in Figure 4, columns are reserved for account names, invoice numbers, and amounts due. Enter some names and amounts there, and the labels shown in Figure 4. If you number your invoices consecutively, you can calculate them. Enter the first invoice number at K20, then at K23 enter:

```
>K23:+K20+1
```

If you replicate that entry relatively into K26, K29, and so forth, then each time a receivable is posted an invoice number that is 1 greater than the preceding number will be produced when the first invoice is posted in K20.

At M19 and N19 the dates that will be calculated will be returned. Since they are already on the sheet at B17 and G17, bring them to this part of the sheet with the entries:

```
>M19:+G17
>N19:+B17
```

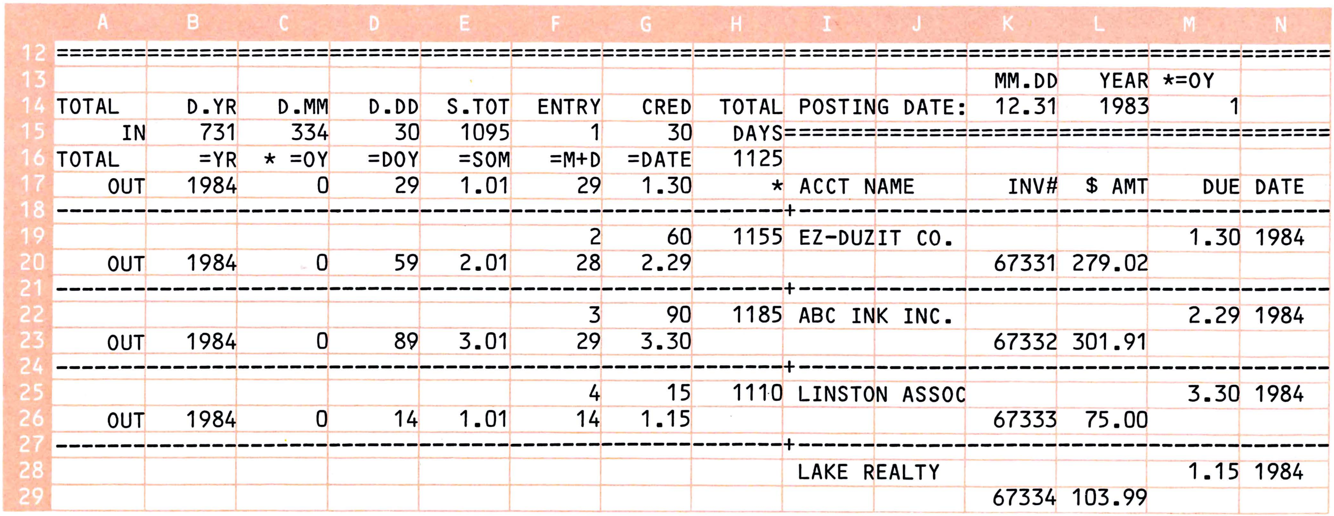
In the event that you do not vary the terms of your billings, replicate those entries with no change:

```
>M19:/R...N19:M22:NN
```

However, if you extend different credit terms to different customers, enter:

```
>M19:/R...N19:M22:RR
```

In that case, your sheet will return zeroes at M22 and N22 because there are no values at the cells to which those entries refer. At least not yet. Having created a range of entries, you can now replicate the due dates down the sheet more efficiently. We'll assume you grant different terms to different customers, but if



```
       A        B        C        D        E        F        G        H        I        J        K        L        M        N
12 ==============================================================================================================================
13                                                                                               MM.DD     YEAR *=OY
14 TOTAL        D.YR     D.MM     D.DD    S.TOT    ENTRY     CRED    TOTAL POSTING DATE:    12.31     1983        1
15        IN      731      334       30     1095        1       30    DAYS======================================================
16 TOTAL         =YR    * =OY     =DOY     =SOM     =M+D    =DATE     1125
17       OUT     1984        0       29     1.01       29     1.30        * ACCT NAME             INV#    $ AMT      DUE DATE
18 ------------------------------------------------------------------------+-----------------------------------------------------
19                                                  2       60     1155 EZ-DUZIT CO.                                1.30 1984
20       OUT     1984        0       59     2.01       28     2.29                           67331   279.02
21 ------------------------------------------------------------------------+-----------------------------------------------------
22                                                  3       90     1185 ABC INK INC.                                2.29 1984
23       OUT     1984        0       89     3.01       29     3.30                           67332   301.91
24 ------------------------------------------------------------------------+-----------------------------------------------------
25                                                  4       15     1110 LINSTON ASSOC                               3.30 1984
26       OUT     1984        0       14     1.01       14     1.15                           67333    75.00
27 ------------------------------------------------------------------------+-----------------------------------------------------
28                                                                          LAKE REALTY                                 1.15 1984
29                                                                                           67334   103.99
```

FIGURE 4. *The first entry on the receivables form is calculated on rows 15 and 17. Subsequent entries are calculated on the rows above the ones on which the entries actually appear. As in Figure 3, note the locations of the cells that require /F$ to display the date correctly.*

not, merely change the final part of the expressions to follow from ":RR" to ":NN". Enter:

```
>M19:/R . . . M22:M25:RR
```

That produces four entries in which the month and day will appear. To expand the year entries, enter:

```
>N19:/R . . . N22:N25:RR
```

Were you now to replicate the ranges M19 . . . M28 and N19 . . . N28 into M31 and N31 respectively, you would double the number of receivables you could post. One further set of replications, and you could double the number again. In short, it takes relatively few keystrokes to greatly expand the amount of working space available.

If you do extend differing lines of credit to different customers, you will have to set up an area of the sheet where you can calculate the differences. Since we have set up the sheet to calculate across rows (/GOR), the most efficient place is to the left and just above the rows where the due dates are needed. That is why we formatted the sheet as we did. The region with I13 as its upper left corner constitutes a rectangle that can be printed, with /PP, as a report. The region beginning at A13 and extending to H13 contains the data and expressions necessary to calculate the dates for the report.

Since the report form we have developed contemplates that all the bills posted will have the same posting date, the calculation of days from the beginning of time is the same for each entry. The only differences to account for are in the days after posting that any bill will fall due. That means that it is only necessary to replicate the entry that adds CRED to S.TOT. Enter:

```
>H16:/R:H19:NR
```

After completing the replication, H19 should show the same number as E15. You can replicate H19 into H22 and then replicate H19 . . . H22 into H25 and so on, doubling the number of entries with each replication.

Now, to calculate the new date in days for differing customers, enter the days of credit for each in column G, cells G18, G21, G24, and so on. Each credit entry will then be in the same relative position as the first with reference to the TOTAL DAYS field.

To break down those days into dates you have to replicate the expressions in B17 through G17. Until you are comfortably adept at reading the expressions on the edit line as you replicate a range of complex expressions from one line into another, it's probably best to replicate one expression at a time. There are, in all, 27 references in cells B17 . . . G17 that must be specified as relative or no change, so we shall take them one at a time. To further avoid confusion as you make the following entries, make sure that G18 contains the same value as G15. Then, as you finish each replication, it should return a value identical to the one being replicated. Enter:

```
>B17:/R:B20:RNN
>C17:/R:C20:RR
>D17:/R:D20:RRNN
>E17:/R:E20:RRNNRNN
>F17:/R:F20:RRRNNRRNN
>G17:/R:G20:RR
```

By repeating that series of replications, beginning with the entries now on row 20 and substituting references to row 23, you can create as many total rows as you need.

Before you actually post anything in the report section of the sheet, go to L14 and either blank the cell or enter 0. Since there is no year 0 in the calendar, the VisiCalc program will return NA at B15, where the count of days begins. If there's a date in K14, C15 and D15 will display values, but because B15 is NA and all remaining calculations depend on that cell, all the remaining cells will show NA, too. (Except M14, the initial leap year calculation, which will show 0 or a blank if it has been formatted with /F∗.)

What you now have on the screen is a very useful tool. Save it as a VisiCalc file, using a filename like POSTFORM. Before saving it, though, scroll the screen until I13 in is the upper left of the window. When the file is loaded later, the report section will appear first. It will be blank, but ready to calculate the dates the moment you enter the current date and the days of credit for each entry. When that has been completed, it can be saved as a DIF file, a VisiCalc file, or printed immediately, as you require. The next time that receivables must be posted, simply reload POSTFORM and begin the process anew.

FOR APPLE /// USERS

The table in our last issue did not contain the command to print out expressions and formatting commands for the APPLE ///.
Here it is:
/SS *.driver name*

Use the same driver name as when using the /PP command.

CONSTRUCTING A SCHEDULING FORM

Schedules are made to be revised, it's said, but using the VisiCalc® program you can create a schedule that, at least in two ways, revises itself.

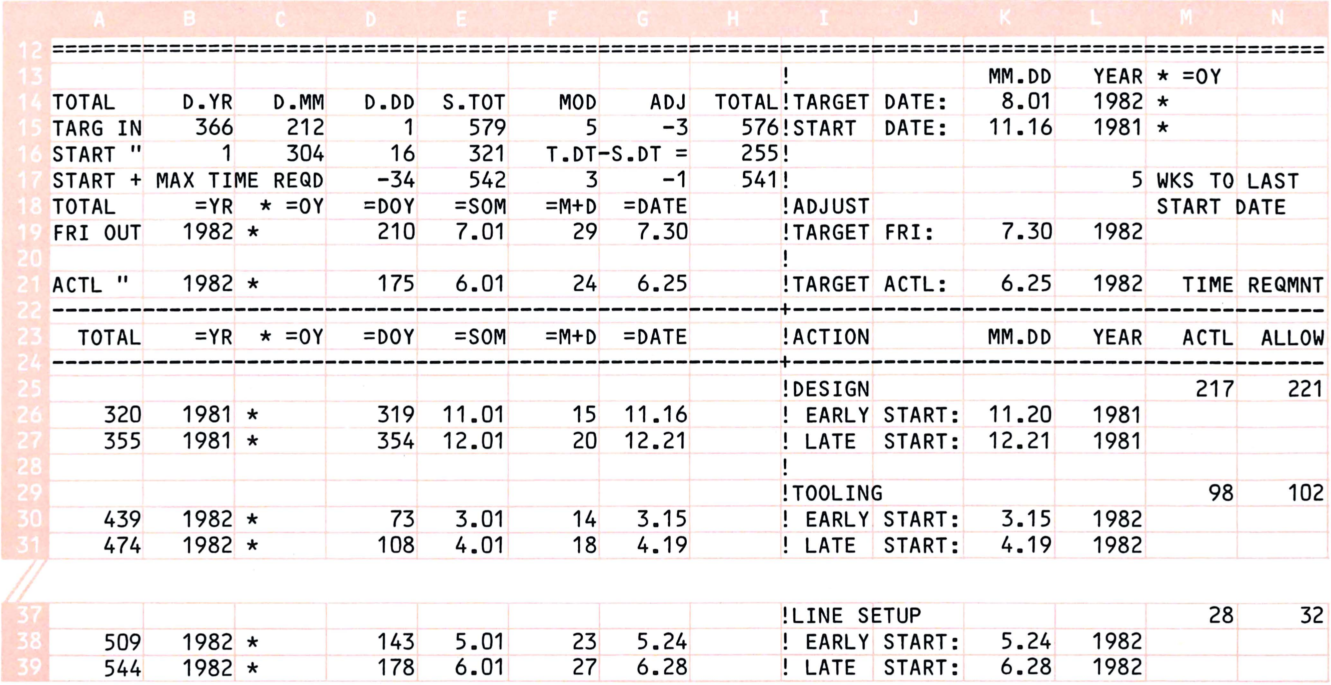
First, we'll set a target date for completing a project, and determine if it is realistic given the time needed to perform the separate steps. Second, we'll establish the dates on which those steps must be begun.

This article is based entirely on the model developed in *General Formula*, but with some added wrinkles. One is a factor that automatically revises the target date to schedule completion on a Friday. Another automatically revises the target date if more time is needed than the target date allows. Next, all start dates are adjusted to Mondays and two start dates (early and late) are shown for each step. Finally, there is a field showing how many weeks there are until the start date.

Our project has only four steps, but our approach can be extended to any number. It will not correctly calculate any schedule that is only one week long.

Calculating The Target Date

Since the sheet we use for this schedule is an extension of the one developed in the *General Formula* article, load the file DATEWORK.



```
      A        B        C        D        E        F        G        H        I        J        K        L        M        N
12 ==============================================================================================================================
13                                                                    !                    MM.DD     YEAR * =OY
14 TOTAL       D.YR     D.MM     D.DD    S.TOT      MOD      ADJ    TOTAL!TARGET   DATE:     8.01     1982 *
15 TARG IN      366      212        1      579        5       -3      576!START    DATE:    11.16     1981 *
16 START "        1      304       16      321   T.DT-S.DT =           255!
17 START + MAX TIME REQD          -34      542        3       -1      541!                                    5 WKS TO LAST
18 TOTAL        =YR    * =OY     =DOY     =SOM     =M+D    =DATE        !ADJUST                              START DATE
19 FRI OUT     1982 *             210     7.01       29     7.30        !TARGET   FRI:      7.30     1982
20                                                                      !
21 ACTL "      1982 *             175     6.01       24     6.25        !TARGET   ACTL:     6.25     1982     TIME REQMNT
22 ---------------------------------------------------------------------+---------------------------------------------------------
23   TOTAL      =YR    * =OY     =DOY     =SOM     =M+D    =DATE        !ACTION             MM.DD     YEAR     ACTL    ALLOW
24 ---------------------------------------------------------------------+---------------------------------------------------------
25                                                                      !DESIGN                                  217      221
26      320    1981 *             319    11.01       15    11.16        ! EARLY  START:    11.20     1981
27      355    1981 *             354    12.01       20    12.21        ! LATE   START:    12.21     1981
28                                                                      !
29                                                                      !TOOLING                                  98      102
30      439    1982 *              73     3.01       14     3.15        ! EARLY  START:     3.15     1982
31      474    1982 *             108     4.01       18     4.19        ! LATE   START:     4.19     1982

37                                                                      !LINE SETUP                               28       32
38      509    1982 *             143     5.01       23     5.24        ! EARLY  START:     5.24     1982
39      544    1982 *             178     6.01       27     6.28        ! LATE   START:     6.28     1982
```

FIGURE 5. *Compare the labels in this figure to those in Figure 4. Although many of the initial calculations are in the same locations, the labels are different.*

Now, edit the file DATEWORK. Enter:

```
>B16:/M . . . B20:/M . . . B20
```

That opens two rows, 16 and 17, leaving the labels at row 18 and the calculations of the new dates in row 19. One expression we will need in two places is already on the sheet at H18, and can be replicated into the proper locations and then deleted. Take care, though, before you enter the final expression listed below, that you have not made any errors entering the first two. The third expression erases the original entry, and is an irreversible step. The only way to recover from an error would be to reload DATEWORK and re-edit it as shown above. Now, enter:

```
>H18:/R:H15:NN
>H15:/R:H17:RR
>H18:/B
```

That writes over the label DAYS at H15, erases H18, and enters the correct expression in H17.

Figure 5 shows the sheet as it should look when completed. The original TARGET DATE is entered at K14 and L14, where POSTING DATE was, so change the labels accordingly. As before, M14 contains the leap year calculation, and is formatted with /F*. At K15 and L15 enter the START DATE shown. Its leap year calculation is at M15, and is replicated from above with the entry:

```
>M14:/R:M15:RR
```

The file DATEWORK contains, in cells B15 . . . E15 the necessary expressions for the beginning of a schedule sheet. One change has to be made, though, at D15, where we subtracted one day to avoid counting the posting date. In a schedule, all days are counted, so the factor −1 must be removed from the expression in D15. You can either enter:

```
>D15:+K14-@INT(K14)*100
```

Or use /E to remove the −1 from the expression.

At E15 is the actual number of days from 1.01 1981 to TARGET DATE. Before referring to a calendar we entered 8.01 1982 as the date. That is a Sunday, so, as shown in Figure 5, at K19, it was adjusted back to 7.30 1982, a Friday.

Adjusting To Fridays

To adjust dates to Friday we used a modulus, or "mod," for short. You've already used a mod when calculating the leap year. It is a calculation that yields as its answer the remainder of a division operation. The expression required to make the adjustment is straightforward, but it will be useful to spell out the process thoroughly. That way, you can adjust to any day of the week you wish.

If we divide S.TOT by 7 we get a decimal remainder that can be converted into whole days by multiplying it by 7. The answer to the multiplication is the one we need, because that tells us which day of the week it is. The general expression is:

Value − (@INT(Value/ n) ∗ n) = Remainder

Where *n* is the number of the units of time. To calculate leap years we used 4, to calculate weeks we use 7. In F15, enter:

```
>F15:/FI+E15-(@INT(E15/7)*7)
```

That expression yields a number from 0 to 6.

January 1st, 1981, where our calendar starts, was a Thursday, so every time the expression returns a 1 that day is also a Thursday. Wednesday is 0, Friday 2, Saturday 3, Sunday 4, Monday 5, Tuesday 6. Because we want the target date to be a Friday we must write an expression that adjusts the excess days to 2. That expression is:

```
>G15:2-F15
```

Then if the result is 2, the expression returns 0 and TARGET DATE, already a Friday, is not changed. If it returns 1, TARGET DATE is a Thursday, and 1 day must be added to the total days. Similarly, if the expression returns 6, the result of 2 – 6 is –4 and four days will be subtracted to adjust TARGET DATE to Friday. The effect is that if TARGET DATE falls between Saturday and Tuesday, it will be adjusted to the preceding Friday. But if TARGET DATE is a Wednesday or Thursday, the following Friday will become TARGET FRI.

The total of days is adjusted with the entry, already on your sheet at H15, which was replicated from H18 when you edited the file DATEWORK. That gives the total days from 1.01 1981 to the Friday closest to TARGET DATE.

The next adjustment that must be made is critical, as it answers the question: Given the target date, the time it takes to get things done, and the proposed start date, what is the actual target date, or TARGET ACTL?

Setting The Actual Target Date

The first step to TARGET ACTL is determining how much time is available from START DATE to TARGET DATE. That requires a calculation of the days to START DATE, which can be entered with /R. Enter:

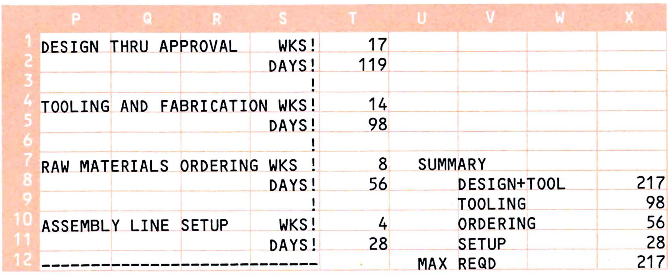
```
>B15:/R:B16:RNN
>C15:/R:C16:RRNNRNN
>D15:/R:D16:RR
>E15:/R:E16:RR
```

That yields the days to START DATE, which we then subtract from the days to TARGET DATE, and is done to the right of the labels T.DT – S.DT =. Enter:

```
>H16:+H15-E16
```

Given the dates we entered on the sheet, that entry shows there are 255 days between START DATE and TARGET DATE.

To compare the time available to the time required calls for subtracting the days available from the days required. Whether the number that is returned is positive or negative, it will be used to determine the earliest date on which we can expect the project to be completed.



```
     P       Q       R       S       T        U       V        W       X
 1 DESIGN THRU APPROVAL     WKS!      17
 2                         DAYS!     119
 3                             !
 4 TOOLING AND FABRICATION WKS!      14
 5                         DAYS!      98
 6                             !
 7 RAW MATERIALS ORDERING WKS !       8    SUMMARY
 8                         DAYS!      56        DESIGN+TOOL          217
 9                             !                TOOLING               98
10 ASSEMBLY LINE SETUP      WKS!       4        ORDERING              56
11                         DAYS!      28        SETUP                 28
12 ----------------------------            MAX REQD                  217
```

FIGURE 6. *The breakdown of the tasks identified in the main part of the scheduling form.*

Beginning at P1 is a list of the four steps in the project, and, at T1, the time each takes in weeks. Beneath the time in weeks is the time in days. Enter the times in weeks as shown in Figure 6, and multiply each by 7 to get the times in days. Enter:

```
>T2:7*T1
>T2:/R:T5:R
```

Then replicate T5 into T8 and T8 into T11.

To the right of the list of steps is a SUMMARY field, showing the actual time requirements. TOOLING is assumed to be dependent on DESIGN, so at X8 enter:

```
>X8:+T2+T5
```

The rest of the entries in SUMMARY are:

```
>X9:+T5
>X10:+T8
>X11:+T11
```

MAX REQD is the maximum of the summary. Enter:

```
>X12:@MAX(X8...X11)
```

Those entries assume that ORDERING and SETUP are relatively independent of TOOLING and DESIGN.

We decided that the project as a whole and its separate steps must begin on Mondays. However, because the original TARGET DATE has been adjusted to a Friday, and the time to complete any step defined in weeks, all start dates calculated from the TARGET DATE will be Fridays, too. To convert them to Mondays, yet allow sufficient time to complete each step, we must subtract a few more days. (That is, we must add them to the time required, and later subtract the total from the TARGET DATE.) Enter:

```
>D17:4+X12-H16
```

That is added to the days from 1.01 1981 to TARGET DATE with the entry:

```
>E17:+H15+D17
```

It remains to convert that total of days into a Friday, which is done by replicating the expressions we used before for the same purpose. Enter:

```
>F15:/R:F17:RR
>G15:/R:G17:R
```

So far only START DATE and TARGET DATE have been expressed as dates. We now have to convert our calculations into dates, too. The editing done earlier, on DATEWORK, has placed the labels and expressions necessary for that in row 19 already. However, the expression at B19 looks up H18, while it should search for H15 in the @LOOKUP table. As above, use /E to change 18 to 15 or enter:

```
>B19:@LOOKUP(H15,B2...M2)
```

The expression now at D19 also contains a reference to H18, so change it accordingly. Edit D19 with /E, or enter:

```
>D19:+H15-@LOOKUP(B19,B1...M1)
```

As now edited, the entries in row 19 give TARGET FRI:. To generate the earliest Friday on which the work can be completed, replicate them into row 21 with the entries:

```
>B19:/R:B21:RNN
>C19:/R:C21:RR
>D19:/R:D21:RRNN
>E19:/R:E21:RRNNRNN
>F19:/R:F21:RRRNNRRNN
>G19:/R:G21:RR
```

Now the compressed dates and years are available to be brought into the fields TARGET FRI:, which defines the Friday nearest TARGET DATE, and TARGET ACTL:, the earliest possible start-up date based on the time available and the time required. Enter those dates in the appropriate fields with:

```
>K19:+G19
>L19:+B19
>K19:/R . . . L19:K21:RR
```

The WEEKS TO LAST START DATE field is calculated simply, since it's just the difference between the time available and the time required. We've written the expression so that if the time available is greater than that required, WEEKS TO LAST START DATE is expressed as a positive integer, and if the time required is greater than that available, a negative integer. The entry is:

```
>L17:/FI-D17/7
```

Scheduling The Tasks

Only a few of the expressions necessary to schedule the various tasks are missing from the sheet. The first is brought to M25 under TIME REQMNT ACTL with the entry:

```
>M25:+X8
```

As above, we have to add four days to adjust to Monday. Enter:

```
>N25:M25+4
```

The time taken for TOOLING is entered at M29 and N29 with:

```
>M29:+X9
>N29:+M29+4
```

The remaining tasks are treated as the TOOLING entry.

Now, at A26, we can schedule the EARLY START of the first task with the entry:

```
>A26:+H17-N25
```

Which subtracts the time required to do the work from the days required for TARGET ACTL.

Then at A27 enter:

```
>A27:+H15-N25
```

Because we now reference slightly different cells for our starting points we cannot replicate rows 19 or 21 directly into row 26. So enter:

```
>B26:@LOOKUP(A26,B2 . . . M2)
>D26:+A26-@LOOKUP(B26,B1 . . . M1)
```

All the remaining cells in that row can be replicated from above. Enter:

```
>C21:/R:C26:RR
>E21:/R:E26:RRNNRNN
>F21:/R:F26:RRRNNRRNN
>G21:/R:G26:RR
```

Then B26 . . . G26 can be replicated into B27, keeping all value references to ranges in the calendar as N (no change) and all other references as R (relative).

Finally, to bring the dates to the right, enter:

```
>K26:/F$+G26
>L26:+B26
```

And then replicate them relatively into the row directly below.

At this stage, all that remains to be done is to replicate rows into rows to make the sheet as extensive as you wish. If you go to K14 and L14 and enter zeroes, the sheet will fill with NA's in most locations. Columns M and N will still show values until you zero the cells to which they refer. Then by scrolling the sheet until the cursor is in the upper left of the window, and at I13, you

VISICALC POINTER . . .

When you are working on a large sheet and have split the screen with the /W command, it can also be useful to employ the /WS command.

That's particularly so if you are using the VisiCalc program as a checkbook register, a general ledger sheet, or any application in which you want to enter or view the results of entries in the same column or row.

The effect of the /WS command is that the motion of the cursor is synchronous for both windows in the direction of the split, but is unsynchronized in the other direction. That is, if you move the cursor vertically after having split the window vertically, both windows will scroll up or down together. If you move the cursor horizontally, however, only the window in which the cursor lies will be affected.

The practical value of the command is that you can use it to move the cursor between windows without changing the row or column (depending on the direction of the split) you were in when you left the other window.

If, for example, you need at M15 a value that was calculated in B15, you would split the window vertically at column C (with /WV), and enter the command /WS. You would then position the cursor on B15, and press the semicolon to move to the second window. The cursor would remain in row 15, and after moving it to M15 you would enter:

```
>M15:+;®
```

When you press the semicolon, the cursor will move to B15 and when you press ®, will return in M15 the value at B15. The same command sequence, +;®, allows you to set up a sheet on which you can distribute the results of earlier calculations or entries to new locations for further analysis or summary.

can save the sheet as a report form into which you can plug various dates to analyze.

With little additional effort you should be able to adapt the approach we have developed in this issue to your own needs. In fact, for some applications you do not have to enter all we have shown. For example, if you wish only to find the number of days between dates, you can stop developing the sheet shown in this article with the entry at H16.

So, as we have shown, if you have a problem that is not of a financial nature, you should not overlook the VisiCalc program as a tool. If the problem can be defined in numerical terms, there's an excellent chance that the VisiCalc program can be used to help solve it.

DOUBLE YOUR NUMBER

Long before there were personal computers it was necessary to multiply two relatively large numbers. If you owned an abacus (a primitive version of VisiCalc®), and knew how to use it, you could slide beads around until you got the product. Lacking an abacus, if you knew the multiplication tables, you could do a long multiplication on a piece of paper.

But what if you didn't own an abacus, and never learned how to use the multiplication tables? Then you could still get the answer using a method known as "peasant" multiplication. All you really needed to know was how to double or halve a number.

To see how it works, load the VisiCalc program, put the multiplicand, 123, into cell A1, and the multiplier, 456, into cell B1. (Peasants would use a piece of paper, but times change.) Now, go to cell A2 and set it to twice cell A1. Enter:

```
>A2:2*A1
```

Then go to cell B2 and set it to half of B1, rounded down to the nearest integer, by entering:

```
>B2:@INT(B1/2)
```

Now replicate those entries down the sheet with:

```
>A2:/R . . . B2:A3 . . . A10:RR
```

Column A now contains a series of doubled numbers, and column B a series of halved numbers that should eventually reach zero.

Next, we need to know which numbers in column B divided evenly by 2, and which did not. The easiest way to do that is to make C1 equal to the difference between B1 and 2*B2 as in the entry:

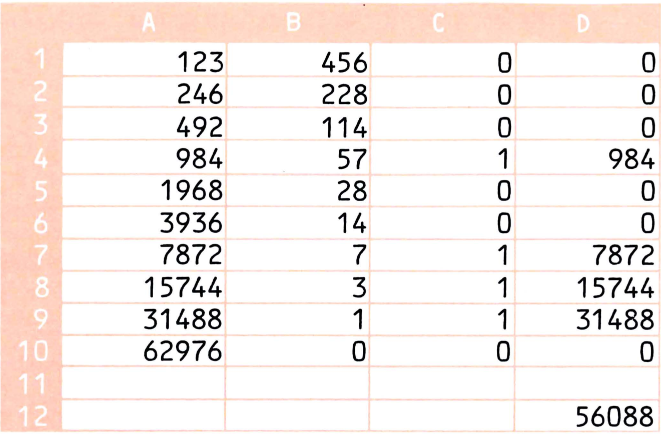
```
>C1:+B1-(2*B2)
```

Since we need that information for each halving, replicate C1 down column C by entering:

```
>C1:/R:C2 . . . C10:RR
```

Now Column C contains only zeroes and ones; a zero if the number halved was even, and a one otherwise.

To multiply, we have to add up, in column D, all of the numbers from column A that are next to an odd number in column B. Those odd numbers in column B are easy to identify, because they are next to 1's in column C, and since the numbers we don't want to include in the addition are next to 0's in column C, all we need to do is multiply the values in column A by the values in column C. So, in column D, enter:



	A	B	C	D
1	123	456	0	0
2	246	228	0	0
3	492	114	0	0
4	984	57	1	984
5	1968	28	0	0
6	3936	14	0	0
7	7872	7	1	7872
8	15744	3	1	15744
9	31488	1	1	31488
10	62976	0	0	0
11				
12				56088

FIGURE 7. *The Peasant Multiplication sheet.*

```
>D1:+A1*C1
```

And replicate it down the screen with the entry:

```
>D1:/R:D2 . . . D10:RR
```

In cell D12 we take the sum of the column by entering:

```
>D12@SUM(D1 . . . D10)
```

The value displayed at D12 is the product of the values you entered in A1 and B1. Try out a few numbers to satisfy yourself that peasant multiplication works, but keep B1 under 1000. If you want to try numbers larger than 1000, you will have to add more rows to the bottoms of columns A-C.

To see how peasant multiplication works, set A1 to 1. Column A now contains the powers of two (2^0 through 2^9), and column C (read from the bottom up) contains the base 2 (binary) representation of the number in B1. Column D contains the powers of two for all of the 1 binary digits so that their sum is the number in B1.

In other words, peasant multiplication works like the multiplier in a modern computer. Since the computer already has its numbers in binary (base 2), dividing by 2 and multiplying by 2 are simple, just as in a decimal (base 10) number system multiplying and dividing by 10 are simple.

SOLUTION TO SEND MORE MONEY

The correct solution to the cryptarithm appearing in the November/December issue is:

```
SENDMORY
95671082
```

That makes:

```
SEND + MORE = MONEY
9567 + 1085 = 10652
```